Building Startup Systems

Objective

1. Setup a working environment for *Python* development
2. Grok the mental model for *Python*
3. Build and deploy a simple working *Python* application
4. Understand best practices + future directions

Get Ready for Python

- Make sure you have the google-cloud-sdk installed
  ```
  $ brew cask install google-cloud-sdk
  ```
- Make sure you have **python 3.6** running locally
  ```
  $ brew install python3
  $ python3 --version
  ```
- Start with a small working Python Project
  ```
  https://github.com/amfleming/pskeleton
  ```
- Install dependencies
  ```
  $ pip3 install -r requirements.txt
  ```
- Install Python Plugin for IntelliJ

Python Basics

```
import logging

def some_function(a1, a2, a3):
    for i in [1,2,3,4]:
        print(i)

class foo:
    x = 5
    def __init__(self):
        self.x += 1
    def baz(self):
        return self.x

if __name__ == '__main__':
    f = foo()
    print(f.baz())
    print(foo.x)
```

References as you Develop

Flask API + QuickStart Guide

https://flask.readthedocs.io/en/latest/api/#api

MongoDB Reference Docs

https://docs.mongodb.com/manual/reference/

MongoDB Best Practices

- **Components** are used to organize and construct the UI
  Smart Components are aware of the data and logic unique to your application. Containers pass data and callbacks as props to presentational components, and handle updating the data when a user interacts with the app.
  Presentational Components are presentation-only, styled components, that defer to their smarter partners for processing. Their only input is their props
- **Views** are primarily used for styling and layout of children elements
- **Styles** configure views - layout, spacing, colors, etc
  see http://www.reactnativeexpress.com/view
- In React it's a convention to export one component from a file, and to export it is as the **default** export.